Presentation

The Bourges Music Software Competition, 1997

Dylan Menzies-Gow, York, UK *rdmg101@unix.york.ac.uk*

LAmb[1], from “Live Ambisonics” , is a single program application written for the Silicon Graphics Indy over the last year. It was originally conceived as a flexible new diffusion performance instrument for the electro-acoustic concert hall, and has since developed other capabilities to aid interactive and algorithmic composition. The novelty of LAmb lies chiefly in its application of ambisonic theory and the actual methods of performance control implemented. This presentation will first provide some background and motivation for the project. A tutorial is then given to guide the new user by demonstrating some of the functions. The accompanying User Guide provides all remaining details. LAmb cannot be properly evaluated without the use of a speaker rig. Fortunately a simple four-speaker horizontal arrangement, fed directly from the amplified Indy outputs, can produce excellent results.

1 Background and Motivation

The author’s interest in diffusion stems from electro-acoustic concerts given at York University Music Department, UK. Here traditional diffusion performance using a mixing desk has been combined with *Ambisonic* joystick panpots. The contrast between the two methods is impressive: The panpots were able to smoothly move sound objects around the audience in a “natural” way, where as the mixing desk seemed to encourage more unstable yet potentially more interesting spatial transformations. After investigating ambisonic theory a little, it was realised that it should be possible to generate ambisonic sound transformations with more variety. The more complex operations required would necessitate a digital rather than an analog approach.

1.1 A Quick Guide to Ambisonics

Ambisonic theory has been in existence for about 20 years, and was developed from a sound psycho-acoustical footing to a remarkably high analytical and practical level by the late Michael Gerzon[2]. It is widely regarded amongst the audio engineering profession as the most suitable basis for making 3-

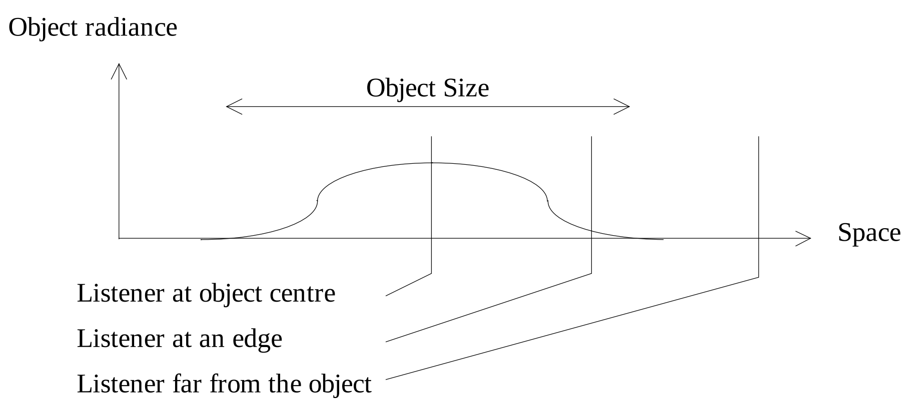
[1] LAmb is freeware and is available on the web from http://www.unix.york.ac.uk/~rdmg101

[2] MA Gerzon, ‘General Metatheory of Auditory Localisation’, Preprint 3306 of the 92nd Audio Engineering Society Convention. For many more references and information about 3d audio see the following web page at York University Music Department: http://www.york.ac.uk/inst/mustech/3d_audio/ambison.htm

dimensional sound *recordings*, particularly in situations where there is no strongly preferred direction or *sound-stage*. In the simplest ambisonic realisation, the soundfield about a listener is approximated by encoding onto only 4 channels collectively know as the *B-format* signal. This can then be *decoded* onto a variety of different speaker arrangements, such that listeners at the centre of the speaker rig perceive a good subjective approximation to the original soundfield. The reconstruction is holographic: The listeners can freely rotate their heads without disrupting the sound scene perceived. The "sweet spot" is very extended which makes ambisonics particularly attractive for concert hall applications. A special microphone called the *Sound Field Microphone* was developed for directly encoding real soundfields into B-format. This is available commercially still, and is a useful tool for the composition of electro-acoustic tape music. Hearing a well made and diffused ambisonic recording is a startling experience, which suggests that it may be possible to diffuse mono sources, live, in a startling way...

1.2 Response to traditional ambisonic pan pot design

Unfortunately analog ambisonic pan pot designs have been lacking in there ability to startle. This is partly understandable due to the difficulties of working with analog electronics, but I believe also partly due to a common notion of the importance of a localised *point* source, rather than a more generalised spatial transformation of a 3-dimensional sound *object*. I found that by using a simple model of an object extended in space, the corresponding calculated B-format signal leads to much more interesting dynamic possibilities in the manner of a flexible swarm which can move around, close to and through the listener, and maintain the characteristic feeling of ambisonic "naturalness". Certainly for a performance system we should wish parameterise spatial sound in an intuitive way. Even compact sources of sound in real life tend to be coupled to there surroundings via resonances and reflections, which expand the point into an object. It is important to appreciate that the perception of the object is really a dynamic process: A continuously perceived stream of events is much more valuable than those events perceived in isolation, as the brain can cross reference information.



A Simple Sound Object Model

The actual shape of the model isn't too important: In any case B-format puts a limit on the possible directional complexity of the encoded signal (That ambisonics works well is a reflection of the fact that directional complexity *of a given signal* is not of *great* importance in perception, although it has been suggested that higher order ambisonics with more channels allowing for greater complexity may be worthwhile.) The Object Size is easily recognised to be of importance: Compare the sound of a fly moving by with the sound of a large vehicle from which closely related vibrations emanate along its length. Contained in the model is a description of the attenuation of sound over distance. In the simplest case of free, lossless 3-dimensional space the energy flux decays as the square root of distance, by conservation of acoustical energy. In real life geometries the signals may be constrained to lower-dimensional paths, and hence even with lossless transmission, the profile of received energy as a function of object position may vary considerably. A range of useful profiles can be selected with the "bumpiness"[3] control described in detail in the tutorial. From a practical musical point of view a particularly useful profile is a "flat" one in which the energy received from the object by the listener is roughly constant: This can used to balance different musical sources that have been previously balanced with mono or stereo systems.

The major deficiency in the simple model is the lack of small spectral variations of directivity. Adding variations in a computationally efficient manner whilst avoiding artefacts characteristic of the process is not easy. However, in most practical concert situations, it seems the room provides a natural crux for the deficiency by adding (high fidelity) variations of its own. Of course a general soundfield could be synthesised by convolving B-format room responses with many notional points making up an object, but this isn't very useful in a performance situation where we need an efficient yet convincing way of rendering a changing object.

A 4 channel delay can be used to spatially enrich a diffused soundfield with minimum cost. The effect is abstract but has subjective qualities akin to early reflections, if the sources move and the delays are short, in the sense that the source becomes surrounded by a haze of related but slightly displaced signals. Longer delays can also be used to good, more abstract, effect.

1.3 Practical Realisation and Control Methods

The Indy is an ideal candidate for the task. It is fast enough to do real time audio processing of the kind desired, although admittedly real time multi-channel fast-convolution would be out of the question. The audio port has 4 high quality inputs and outputs. The four inputs naturally suggest that there should be 4 mono sources simultaneously performable. The audio libraries are well supported and very efficient. The serial port, although not all together reliable, provides a link to MIDI.

The initial conception was to create a MIDI based performance system. Since it was desired that the system should be as openly available for general use as possible, the design was more or less constrained to using the standard MIDI keyboard controller: 5 octaves of velocity-sensitive keys. Traditionally ambisonic pan pots are controlled with two joysticks together, limiting performance by one person to 2 objects. It was therefore interesting to enable the performer to control 4 objects by assigning one object per octave, and thus engage the use of the player's keyboard skills. But how

[3] This term appears a little glib, but was considered to be clear and descriptive to the novice user of LAmb using the graphical interface.

should each object be controlled? Three "modes" of control were developed: In what turned out to be the most useful mode, each key of the octave is related to a position in space, which can be programmed at the start of the performance. When a "position key" is hit the object for that octave begins moving towards the related position with speed determined by the key velocity value - The harder the hit the faster it moves. This has proved to be a simple but very satisfying method of performance, intuitive yet controlled. It is possible to guide an object around quite general paths, without it reaching its destinations, by careful touch and timing. Very slow steady motions are possible of a kind not heard with pan pots or faders. Conversely fast *synchronised* movements of 4 objects are obtainable as well. Provision has been made for keyboards which have joysticks, the Yamaha SY22 being a worthy example. The joystick is also assignable to other functions.

The fifth octave is reserved for soundfield rotation and soundfield sample playback. Soundfield playback was considered to be a useful and novel performance device, as complex soundfields created previously with Csound or by layering outputs from LAmb, could be used. Soundfield rotation is a standard ambisonic technique. The novelty here is that full 3-degrees-of-freedom orientation has been implemented by condensing to a single 3 by 3 matrix operation. Previous analog implementations were limited by complexity to having only 1-degree-of-freedom. The result is that a complex soundfield can be spun around a 3-dimensional rig, giving the listener the strange sensation of head movement but without actually moving the head. The diagram at the head of the User Guide shows how the different elements of LAmb are routed: Sample playback can be rotated, while both objects and samples are combined before applying delay. One can imagine variations of effects achievable by different routings, but it was decided to concentrate on producing a single useful configuration to ease the user's understanding of the system.

1.4 The Graphical Interface, Interactive and Algorithmic Composition

An early version of LAmb was used in a concert before a graphical interface had been developed. It was clear that a GUI would be desirable for configuring the system, and providing a front for saving and loading configuration files. The *Xforms*[4] library was adopted as the least painful way of achieving this. It was natural to provide mouse based control of objects position and attributes, although this was never seen as a substitute for MIDI keyboard performance. In designing the layout and interaction of the interface great attention was paid to making it fast and intuitive to use, much like the worksurface of a modern mixing desk. Also, containing objects in one window greatly reduces overheads incurred by moving between windows. Inevitably the prospect of *interactive composition* was considered, where by the user could layer the movements of several individually recorded objects into one B-format recording. To facilitate this fully, real time soundfile playing and recording functions were incorporated. The program *Mix*[5] from NoTAM is an ideal piece of freeware to complement this process. Within the IRIX environment LAmb and Mix can be used simultaneously without problems.

One of the great advantages of Unix operating systems are the facilities for inter-process communication. Included in LAmb is an option for deriving MIDI data from the standard input stream rather than the physical serial port. This allows additional programs to control LAmb with the minimum of inconvenience. A simple example program, *automidi.c*, has been included to illustrate

[4] Xforms home page is at http://bragg.phys.uwm.edu/xforms

[5] The latest version of Mix can be down loaded from http://www.notam.uio.no/her/fou-e.html

this feature. Potentially the control program could be used to implement some kind of autonomous algorithmic composition. The audio streams themselves can likewise be derived from *named pipes*, although the details of this have not been ironed out in the current version. When this is working the play and record functions will enable full communication with any other processes.

1.5 Future Plans and Conclusion

LAmb has proved to be a popular piece of software at York. Encouragement last year led to the construction of a large, permanent 12 speaker performance facility by the author and Tim Ward of the music department. The object rendering algorithms are central to LAmb, and there is undoubtedly room for their development even within the context of the current processing resources of the Indy. Features such as EQ to model air loss, doppler shift, early reflections and spectral spreading over space could all be implemented with varying degrees of success. The user guide has suggestions for how a dedicated external reverberation processor can be used intelligently within the setup. More processing power would enable internal real time reverb of acceptable quality. The new range of powerful PCs offer exciting possibilities for low cost audio implementations which can reach a much wider audience. Interesting use of real-time fast-convolution on such machines is already possible. While the author's work has moved on to concentrate on other projects, LAmb will certainly be supported for the near future at least.

There are undoubtedly many radically different ways of approaching diffusion. It is the job of the instrument designer to pull together a number of threads which sit together coherently, providing adequate interest for experiment and performance while not submerging the user in over-complexity.

2 A Tutorial For LAmb V0.97

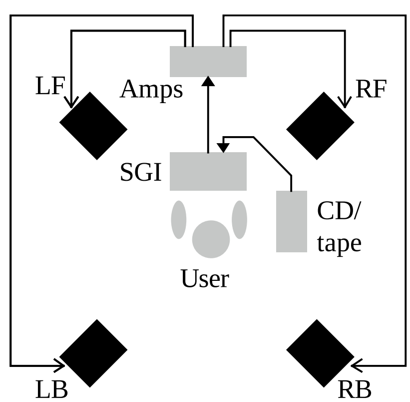
This tutorial will show the user how to set up the simplest possible speaker rig of 4 speakers and use the basic functions of LAmb from the graphical interface. If a MIDI interface and keyboard are available the user can follow the relevant section on keyboard performance at the end.

2.1 Getting Started

First, the pre-requisites:

1. Indy computer, or more powerful SGI fitted with at least 4 channels of audio.
2. Space for four speakers to be arranged in a square around the user sitting centrally with the computer. The ideal width of the square is about 20 feet, but much smaller squares (>5 ft) are acceptable for demonstration purposes. The ambience of the room should be quite subdued but not silent.
3. Four speakers preferably identical with wide and even radiation characteristics. (Most hi-fi speakers are OK.)
4. Two stereo hi-fi amps. Again, these should be the same, but bare in mind LAmb was originally developed on a stereo and two computer speakers! In any case make sure the output levels from the speakers are all balanced, by adjusting the amp volume controls.
5. Connections long enough to link the computer audio outputs to the speakers.
6. A CD or tape player for providing a convenient external audio source for the tutorial.

Use the following plan view as a guide to arrange the equipment:



Now connect the SGI audio outputs 1,2,3,4 to the speaker feeds LF, RF, RB, LB as shown above. On the Indy outputs 1 and 2 come from the mini jack socket next to the arrow out symbol and outputs 3 and 4 come from socket next to the headphone symbol. If you are not sure that your connection is correct or you wish to check, run the csound score and orchestra supplied on disk. The executable *ambi.test* will automatically compile the csound score and load the resulting soundfile into *soundeditor*. The sound test file generated is useful for checking a variety of different ambisonic rigs, and the .orc file contains more help about these. By selecting and repeat playing each of the first 4 seconds, you can check that the speaker connections are sound. (With a microphone held to each speaker, feeding another soundeditor you can check the polarity of connection is correct, as the test waveform is asymmetric. Polarity is unlikely to be confused in this simple set up however.)

The CD or tape player should be connected to input channels 1 and 2. The set-up should now be ready.

2.2 Running LAmb

Transfer the file *lamb* from *Disc 1* to a local hard drive. It will execute immediately: No installation is required. If it does not start then there may well be some kind of hardware or software incompatibility. Unfortunately it has only been possible to compile in IRIX 5.3. *Disk 2* contains an archive file and a executable *comp* which will attempt to make a working version of *lamb:* Transfer both these files to a local directory on your hard drive before executing. If everything works you should be presented with a large window entitled "LAmb.." This window is designed to fit the full size of the smaller of the SGI monitors manufactured. Now carry out the following sequence:

1. Near the bottom left set *Output format* to *Quad: LF, RF, RB, LB*
2. Press the space key or click on the white title bar to switch to the *performance page*.
3. In *Object Locate* the column of sliders on the left reflects the state of object 1. Clicking on the channel buttons at the top switches to other objects. Click on channel 1 and raise the *volume* slider at the top. You can also vary the input level in the configuration page. You should hear the sound from your input source. If not check that input is present with the SGI *audio panel* utility. If input is present you must have a fault somewhere between the output and the speakers.
4. Move the object around by moving the cross hairs in the *Object Locate* section.
5. Now experiment with the other object parameters. Try progressively reducing the *W-adjust* parameter while moving the object, and notice how the quality of the spatialisation changes. Rotate your head while you do this. You will probably find a minimum low value that you prefer. *W-adjust* is present for several reasons. One is to compensate for variations in external decoder design (The master W control on the configuration page can also be used for this.) Another reason is to provide a simple way to optimise localisation according to the spectral composition of the object, without using filters. Finally it is a parameter that can be varied dynamically for special effect, especially at low values where the localisation falls apart completely.
6. With a sensible setting of *W adjust* try varying *Object size* whilst moving the object. At low values you should get the impression of something quite small that moves past you quickly. At higher values it is more like a large swarm of sound is passing by.
7. Now vary *Bumpiness*. At low values the object appears to retreat to a distance. At high values it feels quite claustrophobic as if trapped in a small space.
8. If you have several independent external mono sources, ideally 4, connect these, and manipulate the corresponding objects. Using *Listener Locate* you can "wonder around" the sound scene created by these objects.
9. The *soundfield rotate* control in the bottom right cannot be fully utilised in a horizontal-only set-up, but try it anyway - move the cross hairs horizontally and the object should rotate about you. The *0* button resets this function and saves processing time.

10. Operate the soundfield delay by raising the *Level* sliders within the section. Adjust the time and feedback sliders - the time scale slider is useful for changing the delays whilst not changing their relative relationship. Be careful not to raise the feedbacks to high, especially if raising several. Moving the objects, now leaves moving ghost images. At short delay times, bright room like spatialised reflection results when objects move.
11. Now try the real time file operations. In configuration select a record file name located on a local hard drive, then back in the performance screen hit *record* and move some objects. Hitting record again stops recording. The resulting soundfile can be played directly onto the rig using the *playaiff* command from a shell, or from *soundfiler*. Note that the file format now decoded for 4 speakers and is not B-format.
12. With the *Mix* program referred to earlier, or similar, recorded 4 channel soundfiles can be combined and mixed to produce much more complex soundscapes with more objects. The LAmb *mute* button is useful for silencing LAmb while working temporarily with Mix.
13. To play files directly into LAmb first make up a 4 track file in Mix with independent tracks. In the LAmb configuration page select a play file name in the file section. Now select the *play* button and the chosen file replaces the external audio inputs. Recording can be synchronised with playing by using the small blank button or *ctrl A* . Also note the input file can have 1,2 or 4 tracks.
14. You will be relieved to discover that all the various settings can be saved to a set-up file, from the configuration page. If you save as *init.set* and start LAmb up in the same directory, init.set will automatically reload straight away, and load any samples that were loaded in the last session.
15. Find a refrigerator.

2.3 Using An External MIDI Keyboard

1. The ideal keyboard for this is a Yamaha SY22, but any velocity sensitive keyboard with at least 5 octaves is fine. You should check that the lowest note output is C1 (MIDI note 36.)
2. Connect the keyboard to the SGI via a MIDI interface such as the *MIDI Macman*. On the Indy this must be connected to serial port 2.
3. In the MIDI section of the configuration page set the correct serial port and the MIDI channel which the keyboard is currently sending on. If you have either a Korg M or T series or the SY22 select appropriately with *keyboard type*.
4. In the performance page click on *recall* and select *Quad positions*.
5. With an object present on channel 1, “play” the keys e, f, f#, g in sequence. You should here the object move around clockwise. Play the same keys in different orders and observe how the object moves. Try varying your touch to make the object move with varying speed. You can vary the key response with the *key feel* slider.
6. The 8 “position keys” e to b can be set to different positions by selecting the desired position key, using the cross hairs to position the object then finally clicking on the *copy* button. *Recall* contains some useful presets suitable for different speaker arrangements.
7. Try playing several objects at once using the first 4 octaves.
8. If present, a joystick can be used, but it must first be assigned to an object be playing the d# in octave of that object.
9. The joystick can be used to rotate the soundfield: Play the C in the 5th octave to begin.

10. Key g in the 5th octave and above can be used to play soundfield (B-format) samples. First they must be loaded by clicking on *Load samples* in the config page: Select a directory. LAmb will load files with names of the form 1.aiff, 2.aiff etc to the corresponding keys above g.
11. A number of other performance “modes” exist which you may find useful. Consult the User Guide for more information.

FIN